REMARKS

Claims 1-5 and 21-35 are pending and stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,493,761 to Baker et al. (hereinafter, "Baker"). Claims 1, 3-5, 21, 23-28, and 30-35 have been amended to overcome the objections presented by the Examiner and to provide proper antecedent basis. Applicants believe all claims are in a condition for allowance and respectfully request reconsideration and withdrawal of all objections and all rejections under 35 U.S.C § 102(e).

Without limitation to the claims, embodiments of the present invention relate to a data engine of a Programmable Streaming Data Processor (PSDP) which is arranged to perform primitive functions directly on database records. The data engine processes non-field delineated database records from a mass storage device, such as a disk drive, prior to its being forwarded to a central processing unit (CPU) of a more general processor. The data engine allows the PSDP to perform certain preliminary processing in order to reduce the computational load on the local CPU.

Claims 1 and 21 are independent claims upon which all others depend. Claim 21 is a method analogous the data engine of Claim 1. As recited in independent Claim 1, a data engine receives a non-field delineated data stream from a streaming interface first-in-first-out (FIFO) buffer. The data engine includes a data parser that determines or parses field boundaries in the non-field delineated database records and selects one or more fields to be assembled by an output tuple generator into output tuples. The data engine employs logical arithmetic methods to compare fields with one another, or with values otherwise supplied by general purpose processors, to precisely determine which records are worth selecting to be transferred to memory for further processing by the more general purpose distributed Job Processing Units. The architecture allows for the use of substitution tables, temporary registers, and a data string register to assist in the efficiency and accuracy of the data engine processing.

An output tuple is comprised of the fields of the source record of the disc that are to be selected for further processing by the CPU and PSDP. For example, a record retrieved from disc consists of a record header, typically containing more than one header field, and at least one data field, and typically, many data fields for each record. The collection of fields selected for return to the CPU as a result of processing a record is referred to as a tuple. Possible tuple fields include various record header fields, the PSDP generated record address, unmodified record data fields, a hash field, and tuple status and length information. Boolean results and/or scratch pad words may also form parts of tuples. Most often a tuple will be shorter than the record that was used to generate it, but it may be longer, depending upon the program that is provided to the PSDP.

Baker relates to a data parsing and analysis system. With reference to Figure 1, a logic control module 16 facilitates the input, storage, retrieval, and analysis of data files for displaying or printing the results of the analyses to output devices 18. Storage devices 14 store a data file 20 having at least one protocol header that contains data stored in a plurality of predefined fields. Each data storage device 14 also includes a protocol description 22, that includes a protocol control record and at least one field sub-record, which together describe a subset of a protocol and include rules for analysis of the data protocol subset. A logic control module 16 is capable of retrieving a subset of data from input devices 12 or data files 20 which satisfies one or more criteria based upon extracted field values and filtering criteria contained in one or more of the protocol descriptions 22. The logic control module 16 also includes logic for determining file and protocol header lengths, gathering statistics, determining the next protocol, filtering, and controlling data file manipulation based upon the programmably configurable protocol description. The data input devices 12 may include keyboards, mice, and trackballs. The data output devices 18 may include commercially available computer input/output devices, such as conventional computer displays or printers. Such input and output devices are intended for human interface.

However, the protocol parsing engine of Baker does not process data from a streaming data source prior to its being forwarded to a central processing unit of a more general purpose processor. Further, Baker fails to teach anything that resembles a tuple, let alone an output tuple generator as in the claims. The term "tuple" is used here for the purpose of differentiating raw disc and PSDP output record formats.

In order to anticipate a claim, a reference must teach each and every element of the claim. Thus, Baker does not anticipate the claims because it fails to teach a data engine located in a programmable pipeline processor for processing non-field delineated, streaming, application level database records and an output tule generator as claimed.

With regard to the Examiner's rejection of Claim 1, and Claim 21 which recites a method that may be processed, for example, by the data engine of Claim 1, although Baker indeed teaches a form of a data engine, the data engine taught by Baker is not located in a programmable pipeline processor for processing non-field delineated, streaming, application level database records received from a mass storage device as in the claims. Baker makes no mention of database records, nor any mention of the records being non-field delineated, streaming, or application level. Although Baker does teach a data engine that parses data files, Baker fails to teach a data parser that parses non-field delineated database records. Further, Baker fails to teach parsing the non-field delineated database records into field delineated data. Moreover, there is no suggestion in Baker of an output tuple generator, configured to assemble field delineated data into an output tuple. Although Baker does teach converting files by altering field contents based on the information contained in protocol descriptions, Baker does not teach a tuple generator as claimed. For these reasons, the Examiner's rejection of Claims 1 and 21 is overcome and reconsideration is respectfully requested.

With regard to the Examiner's rejection of Claim 2, Baker fails to teach the claimed programmable memory that serves as a substitution table for field delineated database records. The substitution table enables the filter logic to perform field substitutions as records are processed. An example substitution table may be used in a character data comparison that should ignore the upper and lower case attribute of characters. For example, if a string comparison is to be performed to locate all instances of the word "error" in a field of character text, a substitution table may be used to select instances of the word "Error" as well as instances of "error". To accomplish this, the substitution table maps corresponding ASCII character entries corresponding to capital letters in the ASCII table map to the ASCII code for the corresponding lower case letter, specifically. Thus, if an "E" is presently on the field byte bus and the substitution table is enabled as explained, an "e" would be output by the substitution table. Table 12, as cited by the Examiner, does not teach the claimed substitution table. Rather, Table 12 illustrates character definition lines at the front of a data definition file defining the ASCII character set. Further Claim 2 is dependent on Claim 1 and therefore contains the elements of the base claim. For these reasons, the Examiner's rejection of Claim 2 is overcome and reconsideration is respectfully requested.

With regard to the Examiner's rejection of Claim 22, Baker fails to teach an output tuple generator, let alone anything that resembles a tuple. Further, Claim 22 is dependent on Claim 21 and therefore contains the elements of the base claim. For these reasons, the Examiner's rejection of Claim 22 is overcome and reconsideration if respectfully requested.

With regard to the Examiner's rejection of Claims 23 and 32, Baker fails to teach flagging a record for further processing. Further, Claims 23 and 32 are dependent on Claims 1 and 21, respectively, and therefore contain the elements of each respective base claim. For these reasons, the Examiner's rejection of Claims 23 and 32 is overcome and reconsideration is respectfully requested.

With regard to the Examiner's rejection of Claims 25, 26 and 34, Baker fails to teach identifying the validity of field delineated data by processing an ID field in the header data of the field oriented data. Tables 9 and 10, as cited by the Examiner, list characteristics that might be identified in characters and values defined for those characteristics. They do not, however, teach transaction IDs or identify the validity of field delineated data by processing an ID field in the header data of the field oriented data. Further, Claims 25, 26 and 34 are dependent on Claims 1 and 21, respectively, and therefore contain the elements of the respective base claim. For these reasons, the Examiner's rejection of Claims 25, 26 and 34 is overcome and reconsideration is respectfully requested.

With regard to the Examiner's rejection of Claim 27, Table 12 of Baker illustrates the character definition lines for an ASCII character set. This is not, however, a substitution table. A substitution table can be used, for example, to assist with operations, such as character comparisons whereby upper and lower case distinctions between letters should be ignored. Similar implementations can be used to substitute character sets for language translations and the like. Although the ASCII character set defined by Table 12 could be used in text, it could not be used as a substitution table for performing operations, such as character comparisons. Further, Claim 27 is dependent on Claim 21 and therefore contains the elements of the base claim. For these reasons, the Examiner's rejection of Claim 27 is overcome and reconsideration is respectfully requested.

With regard to the Examiner's rejection of all remaining Claims 3-5, 24, 28-31, 33 and 35, these claims are directly or indirectly dependent upon Claims 1 and 21, respectively, and therefore contain the elements of the respective base claim. For these reasons, the Examiner's rejection of Claims 3-5, 24, 28-31, 33 and 35 is overcome and reconsideration is respectfully requested.

Information Disclosure Statement

An Information Disclosure Statement (IDS) is being filed concurrently herewith. Entry of the IDS is respectfully requested.

CONCLUSION

In view of the above amendments and remarks, it is believed that all claims are in condition for allowance, and it is respectfully requested that the application be passed to issue. If the Examiner feels that a telephone conference would expedite prosecution of this case, the Examiner is invited to call the undersigned.

Respectfully submitted,

HAMILTON, BROOK, SMITH & REYNOLDS, P.C.

By [illegible]

David J. Thibodeau, Jr.
Registration No. 31,671
Telephone: (978) 341-0036
Facsimile: (978) 341-0136

Concord, MA 01742-9133
Date: 4/16/08